Building a Reference Combinatorial Modelfor Dynamic Networks:Initial Results in Evolving Graphs

Afonso Ferreira

INRIA

INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

Building a Reference Combinatorial Model for Dynamic Networks: Initial Results in Evolving Graphs

Afonso Ferreira

N° 5041

December 2003

THÈME 1

ISSN 0249-6399 ISRN INRIA/RR--5041--FR+ENG

SOPHIA ANTIPOLIS

Building a Reference Combinatorial Model for Dynamic Networks: Initial Results in Evolving Graphs*

Afonso Ferreira †

Thème 1 — Réseaux et systèmes
Projet Mascotte

Rapport de recherche n° 5041 — December 2003 — 11 pages

Abstract: Wireless technologies and the deployment of mobile and nomadic services are driving the emergence of complex ad-hoc networks, that have a highly dynamic behavior. Modeling such dynamics, and creating a reference model on which results could be compared and reproduced was stated as a fundamental issue by a recent NSF workshop on networking. In this paper, we show how the modeling of time-changes unsettles old questions and allows for new insights into central problems in networking, like routing metrics, connectivity, and spanning trees. Such modeling is made possible through *evolving graphs*, a simple combinatorial model which helps capture the behavior of dynamic networks *over time*.

Key-words: wireless networks, dynamic networks, routing, energy aware, evolving graphs, graph theoretical models, MANET, fixed-schedule dynamic networks, sensor networks, LEO satellite networks.

* This work was partially supported by the European RTN project ARACNE, the European FET project CRESCCO, the AS CNRS *Dynamo*, and the ACI *Presto*.

† CNRS, Projet MASCOTTE, I3S & INRIA Sophia Antipolis.

Unité de recherche INRIA Sophia Antipolis
2004, route des Lucioles, BP 93, 06902 Sophia Antipolis Cedex (France)
Téléphone : +33 4 92 38 77 77 — Télécopie : +33 4 92 38 77 65

Vers un modèle combinatoire de référence pour les réseaux dynamiques : des résultats préliminaires sur les graphes évolutifs[‡]

Résumé : Les avancées des technologies sans-fil et le déploiement de services mobiles et nomades engendrent l'émergence de réseaux complexes de communication, qui présentent un comportement très dynamique. Récemment, un groupe de travail de la NSF sur les réseaux a déclaré que l'établissement d'un modèle de référence d'une telle dynamique qui permette la comparaison et la reproductibilité des résultats décrits dans la littérature était une question fondamentale dans le domaine. Dans ce rapport, nous montrons que des résultats connus dans les réseaux statiques ne s'appliquent pas aux réseaux dynamiques, et que la modélisation de cette dynamique permet des nouvelles perspectives sur des problèmes centraux dans le domaine des réseaux, tels que les métriques pour le routage, la connectivité, ou encore les arbres couvrants. Une telle modélisation est rendue possible par l'utilisation des *graphes évolutifs*, un modèle combinatoire simple qui aide à cerner le comportement des réseaux dynamiques *au cours du temps*.

Mots-clés : Réseaux sans-fil, réseaux dynamiques, graphes évolutifs, réseaux de capteurs, routage.

1 Introduction

The advent of wireless communication networks, such as ad-hoc, sensor, and LEO satellite networks, highly motivates the study of several facets of their dynamic behavior. The underlying mobility of users and/or relays is just one of the factors that contribute to their dynamics. Others include varying link congestion, node and link faults, and components addition and deletion [12].

Such infrastructure-less mobile communication environments present a paradigm shift from back-boned networks, such as cellular telephony, in that data is transfered from node to node via peer-to-peer interactions and not over an underlying backbone of routers. Naturally, this engenders new problems regarding both optimal routing of data and topology control, under various conditions over these dynamic networks.

Unfortunately, however, most results found in the literature either deal with networks that are static, or are hardly reproducible. This drawback prodded the panel at the NSF Workshop on Fundamental Research in Networking ([11]) to state a recommendation to

> *"Encourage research that seeks radical innovation and paradigm shifts."*,

and

> "*Encourage the development of reference models or benchmarks to ease the burden of reproducible experiments on complex systems.*".

Indeed, as we will see later in the text, established concepts in static networks, like connectivity and spanning trees, must be revisited in dynamic networks. Therefore, we need to envision modeling time-varying properties of MANETS in order to build a combinatorial reference model, and thereby increase reproducibility of networking research outcomes.

This is our research framework in this area, where we focus on the design of models and algorithmic techniques that can harness the complexity of evolving environments.

1.1 This work

In this report, we describe a simple but powerful combinatorial model that captures most characteristics of time-varying networks. The writing will be kept concise and informal in order to highlight the important features and impact of the model. We refer the interested reader to the references given at the end of the paper for a formal presentation of the results shown here.

The notion of *evolving graphs*, introduced recently [8], basically consists in formalizing a time domain in graphs. Surprisingly, this leads to a plethora of interesting questions in algorithmic graph theory, with immediate applications in topology control of mobile Ad-Hoc networks (MANETS), related to verifying properties like optimized routing and connectivity *over time*.

As an example, consider the four snapshots taken at different time intervals of a MANET, as depicted in Figure 1. As one can readily observe, nodes D and G are never connected at a single time interval. Notwithstanding, D can indeed send messages to G, using the *path over time* composed of D, C, E, F, G. Surprisingly, this otherwise trivial fact cannot be directly modeled by usual graphs.

Evolving graphs represent a formal abstraction of dynamic networks, and can be suited easily to this case. Concisely, an evolving graph is an indexed sequence of $\mathcal{T}$ subgraphs of a given graph, where the subgraph at a given index point corresponds to the network connectivity at the time interval indicated by the index number, as shown in Figure 2. The time domain is further incorporated into the model by restricting *journeys* (i.e., the equivalent of *paths over time*) to never move into edges which existed only in past subgraphs.

A journey in an evolving graph is thus a path in the underlying graph whose edge time-labels are in a non-decreasing order. Now, it is easy to see in Figure 2 that D, C, E, F, G is a journey, as mentioned above. Further, note that D, C, E, G is also a journey, with less hops, but delivering the message later (in time interval 3 instead of 2), giving raise to different objective functions that may be optimized. Moreover, notice that although the communication network in Figure 1 is symmetric, the hop-distance over time from D to G ($=3$) is different from G to D ($=4$). In the remainder, we will see that this is just one of the usual concepts that have to be formally revisited under the light of "time" modeling.

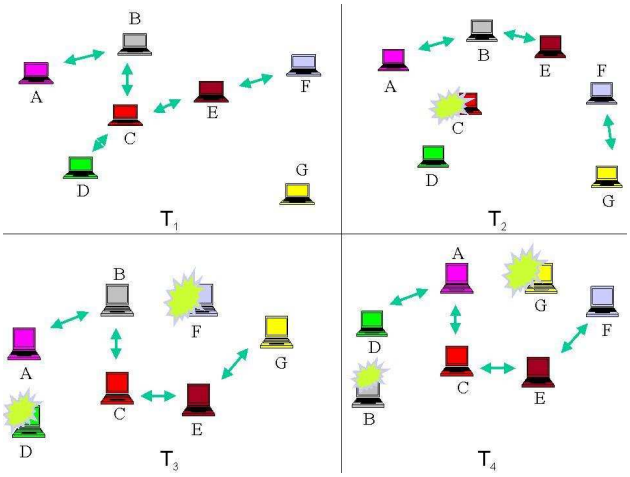

Figure 1: The evolution of a MANET over time. The indices correspond to successive snapshots.

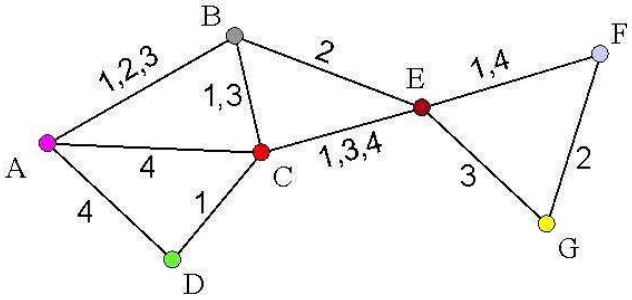

Figure 2: Evolving graph corresponding to the MANET in Figure 1. Edges are labeled with corresponding presence time intervals. Observe that E, G, F is not a valid journey, since the edge {G,F} exists only in the past with respect to {E,G}.

This paper is organized as follows. The definitions of evolving graphs and of some of their main parameters are given in the next section. Then, in Section 3, we show that old concepts in static networks give rise to surprising new insights in dynamic networks, implying that in order to achieve a combinatorial reference model for wireless networks we need indeed to yield substantial innovation and to accept paradigm shifts. In Section 4, we show how this new paradigm can help solving important questions in MANETS, with applications touching energy-aware topology control, and routing. Finally, we close the paper with concluding remarks and several ways for further research in this very little explored field.

2 A graph theoretical model grasping network evolution *over time*

As shown in Figure 2, an *evolving graph* could be seen as a network $\mathcal{G} = (V_\mathcal{G}, E_\mathcal{G})$, where each node and link has a presence schedule defined for it, as follows. Corresponding to each link e in $E_\mathcal{G}$ (respectively, node v in $V_\mathcal{G}$) the *link presence schedule* (respectively, *node presence schedule*) indicates the time instants in which the link (respectively, the node) is present, and possibly other parameters they take during each interval. In the results presented in this paper, both presence schedules are supposed to be given as a sorted list of intervals. Let also $|V_\mathcal{G}| = N$ and $|E_\mathcal{G}| = M$.

For most of the issues discussed in this paper, the informal definition given above suffices. For analytical purposes only, we give below the formal definition of the model.

Definition 1 *Let be given a graph $G(V,E)$ and an ordered sequence of its subgraphs, $\mathcal{S}_G = G_1, G_2, \ldots, G_\mathcal{T}$ such that $\bigcup_{i=1}^{\mathcal{T}} G_i = G$. Let $\mathcal{S}_T = t_0, t_1, t_2, \ldots, t_\mathcal{T}$ be an ordered sequence of time instants. Then, the system $\mathcal{G} = (G, \mathcal{S}_G, \mathcal{S}_T)$, where each G_i is the subgraph in place during $[t_{i-1}, t_i[$, is called an* evolving graph. *We denote $|V| = N$ and $|E| = M$. We let $V_\mathcal{G} = V$, and $E_\mathcal{G} = E$.*

2.1 Network activity and dynamics

With the help of the edge and node schedules, a measure of how much an evolving graph changes its topology can be defined. First, the *activity of a vertex* v can be defined as the number of its presence intervals, and denoted $\delta_V(v)$, and the *activity of an edge* e can be defined as the number of its presence intervals, and denoted $\delta_E(e)$. Then, the *vertex activity* of an evolving graph is defined as the maximum of the vertex activities $\delta_V = \max\{\delta_V(v),\ v \in V_\mathcal{G}\}$, and the *edge activity* as the maximum of the edge activities $\delta_E = \max\{\delta_E(e),\ e \in E_\mathcal{G}\}$. The *activity of an evolving graph* is then defined as the maximum between the node and the edge activities $\delta = \max(\delta_V, \delta_E)$. Finally, the *dynamics of an evolving graph* can be defined as $\frac{(\delta-1)}{\mathcal{T}}$. As a consequence, since usual graphs have $\delta = 1$, they have dynamics zero.

2.2 Paths over time: Journeys

The time domain is further incorporated into the model in the definition of *journeys*, as follows. Let R be a path $R = e_1, e_2, \ldots, e_k$ with $e_i \in E_\mathcal{G}$. Let $R_\sigma = \sigma_1, \sigma_2, \ldots, \sigma_k$ be a time schedule indicating that edge e_i is to be traversed at time σ_i. We define a *journey* $\mathcal{J} = (R, R_\sigma)$ if and only if R_σ is in accordance with R and $\mathcal{G}$, i.e., $\mathcal{J}$ allows for a traversal over time from u to v in $\mathcal{G}$.

It is important to notice that journeys connect two nodes over time, even in the case the nodes are never connected in each configuration, as seen above.

Traversal times for the links, representing the duration of transmitting one packet over a link in the network may also be defined. To simplify computations, it is assumed, without loss of generality, that the intervals of presence of each edge e are closed and that they are longer than the time required to traverse the edge.

Several other parameters may be defined over the evolving graphs. Such parameters, like link traversal time, and node or link costs, may or may not vary over time.

3 Old concepts, new insights

Interestingly, some old concepts in network modeling must be revisited when studying properties *over time*. These include, for instance, the notions of paths, distance metrics, connectivity, and trees, as it will be shown in this section.

3.1 Journey metrics

As far as metrics are concerned, several factors may be seen as important to be optimized when designing "shortest" journeys algorithms. Imagine, for instance, that one application requires messages to be delivered as soon as possible. Another application may be energy-aware, and require that message delivery uses as few hops as possible. Or yet, the objective could be to minimize the delivery delay – *i.e.* the time spent by the message within the network, once it is injected.

The Evolving Graph model allows very easily to take such different metrics into account. We shall exemplify this feature with algorithms that find single source *foremost*, *min-hop*, and *fastest* journeys to all other nodes in the network, which find, respectively, the earliest arrival date, the minimum hop-number, and the minimum delay message delivery strategies. Their basic techniques are described in the following.

Foremost journeys. The problem of computing foremost journeys (i.e., journeys that arrive the earliest possible) from a source node s to all other nodes, has been studied several times in the literature, as shortest time paths in time-dependent transport networks (see [10] and references therein). The Evolving Graph model allows for a presentation with a formal and detailed complexity analysis, as follows.

Remind that, in order to compute shortest paths, the usual Dijkstra's algorithm [6] proceeds by building a set C of *closed* vertices, for which the shortest paths have already been computed, then choosing a vertex u not in C whose shortest path estimate, $d(u)$, is minimum, and adding u to C, i.e., closing u. At this point, all arcs from u to $V - C$ are *opened*, i.e., they are examined and the respective shortest path estimate, d, is updated for all end-points. In order to have quick access to the best shortest path estimate, the algorithm keeps a min-heap priority queue Q with all vertices in $V - C$, with key d. Note that d is initialized to ∞ for all vertices but for s, which has $d = 0$.

The main observation in Dijkstra's method is that prefix paths of shortest paths are shortest paths themselves. Unfortunately, it is obvious that a prefix journey of a foremost journey is not necessarily a foremost journey. Notwithstanding, it can be proven that there exists at least one foremost journey with such a property in an evolving graph.

Therefore, the algorithm for foremost journeys, sketched below, can be seen as a direct adaptation of Dijkstra's technique.

1. Delete root of heap into x.
2. For each open neighbor v of x do
   (a) Compute first valid edge schedule time greater or equal to current time step
   (b) Insert v in the heap if it was not there already.
3. If needed, update distance to v and its key.
4. Update the heap.
5. Close x. Insert it in the foremost journeys tree.

A foremost journey from a source node s to all others nodes can thus be computed in $O(M(\log \delta_E + \log N))$ time. The term $\log \delta_E$ stems from the lookups into the schedule list of intervals, required to decide the earliest time interval in which to cross each visited edge.

Min-hop journeys. The difficulty to compute min-hop journeys stems from the parameter measuring the edge traversal times, which again make that prefix of shortest journeys are not necessarily shortest (see Figure 3).

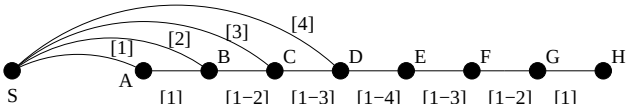

Figure 3: The min-hop journey from S to H takes 8 hops at time step 1, whereas the shortest journey to D takes only one hop, but at time step 4.

Nevertheless, we note that if the last edge, say (u, v), of a min-hop journey between vertex s and vertex v arrives at time t, then the prefix journey (going from s to u) is shorter than all the journeys from s to u ending *before* t. Therefore, one can consider certain pairs (u, t) and compute the min-hop journeys from s to vertex u arriving *before* time t. In this manner, the prefix property is respected, that is, a prefix of a min-hop journey will have a minimum number of hops itself, under the condition that it arrives before some time step t'. Using this property, a tree of journeys between s and pairs (u, t) can be built, in which each vertex u appears at least once (see Figure 4).

Such an algorithm computes min-hop journeys from a single source s to all the vertices in $\mathcal{G}$, if such journeys exist. If $\mathcal{G}$ is connected, then the complexity of the algorithm is $O(Md\ log\delta_E)$, where d is the eccentricity of s. If $\mathcal{G}$ is not connected the complexity of the algorithm is $O(NM\ \log \delta_E)$.

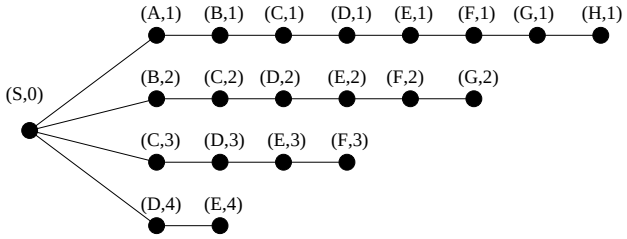

Figure 4: Tree of shortest paths of the evolving graph shown in Figure 3.

Fastest journeys. Here, the objective to be optimized is the journey *delay*. This question is much more complex than the two former ones, because a faster journey may appear well ahead in time, or can have many more hops compared to the min-hop journeys. Moreover, the speed of a journey prefix is almost irrelevant regarding the speed of the whole journey. Indeed, a fast prefix may well imply a long waiting time in a node, offsetting the apparent gain in speed. On the other hand, some prefix journeys are too slow and hence useless to the computations.

As in the case of min-hop journeys, the algorithm proceeds hop by hop, since the number of hops in a fastest journey is also bounded by N. For each k, a list of relevant journey classes of length k starting in s is built, by taking the list for hop-count $k-1$ and extending its relevant journey classes to k hops. This is done by examining each edge of the evolving graph, building the journeys classes that can go through this edge, and then eliminating irrelevant journey classes. After N hops, the fastest journeys belong to the final list of relevant journey classes. Therefore, it suffices to search for the minimum journeys in this list to obtain the requested fastest journeys. The number of relevant classes is bounded by the size of $\mathcal{G}$, and thus the complexity of the algorithm remains bounded.

The hop-count is used to stop the algorithm (after hop-count $N-1$). Track of relevant journeys is kept as follows. Given two journeys $\mathcal{J}_1$ and $\mathcal{J}_2$ from s to v, with departure dates $t_{departure_1}, t_{departure_2}$ and arrival dates $t_{arrival_1}, t_{arrival_2}$, observe that if both $t_{departure_1} \geq t_{departure_2}$ and $t_{arrival_1} \leq t_{arrival_2}$ hold, then not only journey $\mathcal{J}_1$ starts after journey $\mathcal{J}_2$, but journey $\mathcal{J}_1$ arrives before journey $\mathcal{J}_2$. In this case journey $\mathcal{J}_2$ is useless for the algorithm, which keeps track only of journey $\mathcal{J}_1$.

The algorithm computes fastest journeys from a single source s to all the vertices in $\mathcal{G}$ in $O(NM^2\delta_E)$ steps.

Summary Modeling time in MANETS give raise to several different metrics that may serve as objective functions in routing strategies. Here we showed how to optimize the arrival date, the hop-count, or the delay of message delivery. These three parameters can be optimized in polynomial time.

3.2 Connectivity over time

Computing a maximal connected component in a usual graph is very easy [6]. But what would happen when the notions of paths and connections are to be understood *over time*? Let us start by defining time-connectivity formally.

An evolving graph $\mathcal{G}$ is said to be *time-connected* if there exists journeys in $\mathcal{G}$ between any two vertices in $V_{\mathcal{G}}$.

Definition 2 *A* maximal time-connected component *(MTCC) in an evolving graph is the maximal set of vertices $U_{\mathcal{G}} \subseteq V_{\mathcal{G}}$ such that for any pair $u, v \in U_{\mathcal{G}}$, there exists a journey from u to v and a journey from v to u using only vertices in $U_{\mathcal{G}}$.*

Thus, the evolving subgraph $\mathcal{G}'$ induced by considering vertices in the MTCC $U_{\mathcal{G}}$ is a maximally connected evolving graph. For example, in Figure 5, $\{a, b, c\}$ forms a MTCC since there are journeys from a to c and vice versa which traverse only vertices in the set $\{a, b, c\}$.

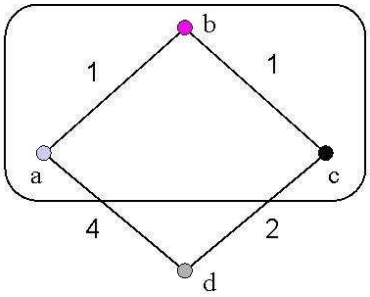

Figure 5: Maximal Time-Connected Component.

On the other hand, in Figure 6, although there exist journeys from a to c and from c to a, $\{a, c, d\}$ is not an MTCC since the only journey from a to c traverses via b. Indeed the subgraph induced by $\{b, c\}$ is not maximally connected, according to the definition above. This is because, unlike standard graphs, there can be a journey between two vertices in the MTCC that traverses vertices outside $U_{\mathcal{G}}$. Thus, it is possible for two vertices $u, v \in U_{\mathcal{G}}$ to establish a journey between them without the constraint that all vertices in the journey must be within $U_{\mathcal{G}}$. So, we offer a looser definition of connectivity as follows.

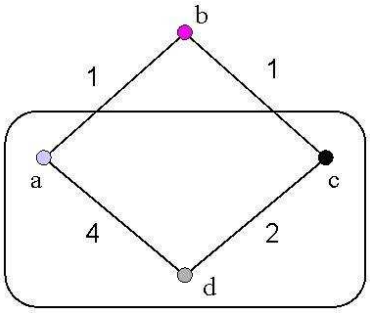

Figure 6: Open MTCC.

Definition 3 *An* open maximal time-connected component *(o-MTCC) is the maximal set of vertices $U_{\mathcal{G}} \subseteq V_{\mathcal{G}}$ such that for any pair $u, v \in U_{\mathcal{G}}$, there exists a journey from u to v and a journey from v to u in $\mathcal{G}$.*

At first sight, it may seem odd that evolving graphs admit two different notions of time-connected components. This stems from the implicit order introduced by the time factor, which allows for a vertex to belong to two different MTCCs, in opposition to the case in usual graphs.

From the computation complexity viewpoint, the fact that computing journeys is polynomial implies that the verification of time-connectivity is polynomial. On the other hand, it can be proven that the decomposition of an evolving graph into (o-)MTCCs is NP-Complete, by reducing the *CLIQUE* problem to *T-COMPONENT*, defined below.

T-COMPONENT: Given an evolving graph $\mathcal{G} = (V_{\mathcal{G}}, E_{\mathcal{G}})$ and an integer k, is there a MTCC of size k?

Summary The results presented above imply that computing a maximal connected component over time of a MANET is NP-Complete. Furthermore, two definitions of such components are possible, one where all connecting journeys belong to the component, and another where some nodes outside the component can be used to ensure time-connectivity.

3.3 Evolving trees

The very notion of *tree* is not immediately translatable when modeling time in MANETS. This is because trees in usual graphs represent a minimum structure capturing connectivity, which is already a complex notion when understood over time, as seen in the previous section.

In order to formally define trees over time, we need first to introduce the broader concept of subgraphs of an evolving graph, or *sub-evolving graphs*. Intuitively, a sub-evolving graph is an evolving graph in which the presence intervals are contained in the presence intervals of the original evolving graph, meaning that a sub-evolving graph can ensure only a subset of the communications possible in the original evolving graph.

Definition 4 *Let $\mathcal{G} = (G, \mathcal{S}_G, cS_T)$ be an evolving graph. $\mathcal{G}' = (G', \mathcal{S}'_G, \mathcal{S}'_T)$ is called a sub-evolving graph of $\mathcal{G}$ if G' is a subgraph of G and if there is a monotone function $f : \mathbb{N} \to \mathbb{N}$ such that for all i, G'_i is a subgraph of $G_{f(i)}$ and $[t_{i-1}, t_i[\subset [t_{f(i)-1}, t_{f(i)}[$.*

Observe that even if a sub-evolving graph allows less communication, its dynamics and space complexity may be greater than those of the original evolving graph, because the schedule lists may be broken into a larger number of intervals.

Therefore, the definition of *rooted evolving trees*, given below, is such that it guarantees that they have minimal space complexity.

Definition 5 *Given a vertex $r \in V$, an evolving tree in $\mathcal{G}$ with root r is a sub-evolving graph $\mathcal{G}_r \subset \mathcal{G}$, such that G_r is a tree, such that each edge has* exactly one *presence interval, and such that there is a journey from r to every vertex in the tree. We say that a rooted evolving tree is* spanning *if it contains all vertices in V.*

Thus, an evolving tree rooted in r allows broadcasting from r, has edge activity 1, and minimal space complexity.

The minimum broadcast time for a source r in an evolving graph is defined as the minimum time t such that there would still be a journey from s to every vertex, should the evolving graph be truncated after time t. A minimum-time broadcast tree is then a spanning evolving tree rooted in r that has the same broadcast time as the original evolving graph. This tree is evidently composed of foremost journeys.

This is particularly interesting because it means that rooted evolving trees are always part of a sub-evolving graph that allows communication from a single node. From this, we know that optimal structures for broadcasting are rooted spanning evolving trees, provided that the cost function respects the inclusion property, *i.e.*, the cost of a sub-evolving graph is smaller than the cost of other sub-evolving graphs that contain it.

Minimum spanning evolving trees Given a cost function $c : E \to \mathbb{R}_+$ on the links of the network, let the cost of a broadcast structure consist of the sum of the costs of each link used in the structure.

This cost function on the structure respects the inclusion property, defined above, so the minimum cost broadcast structure on an evolving graph will be a rooted spanning evolving tree.

Unfortunately, although minimum-cost spanning trees can be found in polynomial-time in standard graphs [6], the problem of finding a minimum-cost rooted spanning evolving tree was proven to be NP-hard, by a reduction from the Steiner tree problem. In the Steiner problem, there are given a planar graph $G = (V, E)$, and a set of vertices $X \subset V$. The problem consists in finding a tree in G containing all the vertices in X, and such that the sum of the costs of its edges is minimum.

The related problem of minimizing the maximum cost in a rooted spanning evolving tree can be solved in polynomial time by an algorithm that computes M times a new Dijkstra tree. This algorithm allows to store intermediate information on the network, so best paths towards any node can be computed independently . If independent paths are not important, one can make a binary search on the maximum cost, erase all edges above the cost and compute a tree. With this option, the worst case complexity drops to $O(\log M \times (M + N(\log N + \log \delta)))$.

Summary One way of defining the notion of trees over time was discussed here. Rooted trees over time were defined, and two minimum such trees were addressed: Minimum-cost rooted spanning evolving trees, and rooted spanning evolving trees which minimize the links of maximum cost. Computing the former is NP-Complete, while computing the latter is polynomial.

4 Applications to energy aware topology control of MANETS over time

We briefly describe some applications of evolving graph modeling to central questions in wireless networking.

Energy efficient broadcast trees over time Under the assumption that the nodes in an ad-hoc network can adjust their transmission power on demand, one particularly widely studied question is as follows. Given a source in the network, find a power assignment for each node such that the total amount of energy required by the system to broadcast a packet is minimized. This problem is known as the *minimum-energy broadcast routing* (MEBR) problem [3, 14], and was proved to be NP-Hard in [5] and several more times since then.

Two heuristics were proven to achieve a 12-approximation for MEBR [13], namely the *Broadcasting Incremental Power* (BIP) and the Minimum Spanning Tree (MST)-based, presented in [14]. The analysis of both heuristics assumed planar networks with *no mobility*, that were modeled as *usual graphs*.

However, the results shown in the previous section imply that in a setting where even a small amount of dynamics is allowed to the network, then computing the equivalent of a MST over time is itself NP-Hard. Therefore, the MST-based heuristic is not an alternative.

As a positive note, we remind that, as pointed out in [4], when the nodes energy is finite, the energy may be the real hard constraint to be met, and the maximum life-time of the network should be the target of communication schemes. The polynomial-time algorithm presented in the previous section builds a rooted spanning evolving tree of the network, that minimizes the maximum energy used by any one node. Such tree could then be used in order to maximize the life-time of wireless communication networks with fixed communication schedules, like sensor networks.

This constraint impacts also strategies for routing over time, as described below.

Power aware routing over time In networks where nodes have small batteries, a node taking part in a routing operation may have its memory emptied when it disappears. This implies that in case of reappearance, the node may have lost the received messages, although it should continue serving as routing relay.

In this case, the time for computing journeys increase by a factor proportional to the node activity, but remains polynomial. In particular, computing foremost and shortest journeys cost now $O(M\delta_V\ (log\delta_E + logN))$ and $O(Md\delta_V log\delta_E)$ time, respectively, whereas the complexity of the algorithm for fastest journeys, increases to $O(NM^2\ \delta_V^2\delta_E)$ time steps.

Other applications include the utilization of the journeys strategies presented in this paper, in the design of routing strategies minimizing different metrics for Fixed Schedule Dynamic Networks, like sensor and LEO satellite networks.

5 Perspectives

The theory of Evolving Graphs is in its infancy, as it is the case with Wireless Networks Theory as a whole, according to [11]. In that report, it is expressed the need for reproducibility of research outcomes through the development of reference models, as well as the fact that these new autonomous networks will be harnessed only through substantial innovation and paradigm shifts. The first few results on Evolving Graphs show their interest and relevance as a combinatorial tool for modeling some dynamic networks [1, 9, 2]. Other related results can be found in [7, 10], for instance. The way is wide open to motivated researchers for building a new combinatorial and algorithmic toolkit for such networks.

References

[1] S. Bhadra and A. Ferreira. Complexity of connected components in evolving graphs and the computation of multicast trees in dynamic networks. In S. Pierre, M. Barbeau, and E. Kranakis, editors, *Proceedings of Adhoc-Now'03*, volume 2865 of *Lecture Notes in Computer Science*, pages 259–270, Montreal, October 2003. Springer Verlag.

[2] B. Bui-Xuan, A. Ferreira, and A. Jarry. Computing shortest, fastest, and foremost journeys in dynamic networks. *International Journal of Foundations of Computer S cience*, 14(2):267–285, April 2003.

[3] W.T. Chen and N.F. Huang. The strongly connecting problem on multihop packet radio networks. *IEEE Transactions on Communications*, 37(3):203–295, oct 1989.

[4] I. Chlamtac, M. Conti, and J.J.N. Liu. Mobile ad hoc networking: imperatives and challenges. *Ad Hoc Networks 1*, pages 13–64, 2003.

[5] A. Clementi, P. Crescenzi, P. Penna, G. Rossi, and P. Vocca. On the complexity of computing minimum energy consumption broadcast subgraphs. *18th Annual Symposium on Theoretical Aspects of Computer Science, LNCS 2010*, pages 121–131, 2001.

[6] T. Cormen, C. Leiserson, and R. Rivest. *Introduction to Algorithms*. The MIT Press, 1990.

[7] A. Faragò and V.R. Syrotiuk. MERIT: A unified framework for routing protocol assessment in mobile ad-hoc networks. *Proceedings ACM Mobicom*, pages 53–60, 2001.

[8] A. Ferreira. On models and algorithms for dynamic communication networks: The case for evolving graphs. In *Proceedings of* 4^e *rencontres francophones sur les Aspects Algorithmiques des Télécommunications (ALGOTEL'2002)*, pages 155–161, Mèze, France, May 2002. INRIA Press.

[9] A. Ferreira and A. Jarry. Complexity of minimum spanning tree in evolving graphs and the minimum-energy broadcast routing problem. Technical report, INRIA, 2003.

[10] B. Kotnyek. An annotated overview of dynamic network flows. Technical Report 4936, INRIA, 2003.

[11] National Science Foundation, USA. *Report of the National Science Foundation Workshop on Fundamental Research in Networking*, 2003.

[12] I. Stojmenovic, editor. *Handbook of Wireless Networks and Mobile Computing*. John Wiley & Sons, February 2002.

[13] P.J. Wan, G. Calinescu, X. Li, and O. Frieder. Minimum energy broadcast routing in static ad hoc wireless networks. *Wireless Networks*, 8:607–617, 2002.

[14] J. Wieselthier, G. Nguyen, and A. Ephremides. On the construction of energy-efficient broadcast and multicast trees in wireless networks. In *Proc. IEEE Infocom*, pages 585–594, Tel Aviv, 2000.

Unité de recherche INRIA Sophia Antipolis
2004, route des Lucioles - BP 93 - 06902 Sophia Antipolis Cedex (France)

Unité de recherche INRIA Futurs : Parc Club Orsay Université - ZAC des Vignes
4, rue Jacques Monod - 91893 ORSAY Cedex (France)
Unité de recherche INRIA Lorraine : LORIA, Technopôle de Nancy-Brabois - Campus scientifique
615, rue du Jardin Botanique - BP 101 - 54602 Villers-lès-Nancy Cedex (France)
Unité de recherche INRIA Rennes : IRISA, Campus universitaire de Beaulieu - 35042 Rennes Cedex (France)
Unité de recherche INRIA Rhône-Alpes : 655, avenue de l'Europe - 38334 Montbonnot Saint-Ismier (France)
Unité de recherche INRIA Rocquencourt : Domaine de Voluceau - Rocquencourt - BP 105 - 78153 Le Chesnay Cedex (France)

Éditeur
INRIA - Domaine de Voluceau - Rocquencourt, BP 105 - 78153 Le Chesnay Cedex (France)
http://www.inria.fr
ISSN 0249-6399